

Manifold-Kernels Comparison in MKPLS for Visual Speech Recognition

Amr Bakry, Ahmed Elgammal

Rutgers Univesity

ambakry@cs.rutgers.edu, elgammal@cs.rutgers.edu

Abstract

Speech recognition is a challenging problem. Therefore, considering the visual information besides the acoustic stream, is essential for improving the recognition accuracy in real-life un-constraint situations. MKPLS is shown to be successful framework for solving visual speech recognition. However, MKPLS lacks in-deep analysis to leverage its power. This work is intended to study two core building blocks in the pipeline of MKPLS: manifold parameterization and manifold kernel. For manifold parameterization, we study the effect of changing the number of centers and the regularization factor in order to find the best parameters of this step. Quantifying the similarity between visual units is challenging step for computing the kernel. Therefore, we explore three kernel categories: matrix-based, curve-based and subspace-based kernels. Intuition behind each kernel choice is provided and quantitative comparison among them is conducted. In other words, this study is intended to reason about the kernel choice for VSR. We compare different kernels using MKPLS framework. We use two public datasets: OuluVs and AvLetters databases.

1. Introduction

Audio visual speech recognition (AVSR) has been investigated intensively in the last few decades [1]. Specially after bimodal fusion of audio and visual stimuli in perceiving speech has been demonstrated by the *McGurk* effect [2]. For example, when the spoken sound /ga/ is seen as /ba/, most people perceive the sound as /da/ [2]. More specifically, with the advances in computer vision, visual speech recognition (VSR), also called lipreading, have attracted research attention [3]. VSR systems gain importance with the need for controlling machines verbally in noisy acoustic environment. An example of that is the car environment, where the noise (e.g. from motor and radio) makes it very difficult for audio speech recognition. Another potential example is to control robots in outer space where there is no media for audio transmission.

Mapping between phonemes and visemes¹ tends to be many-to-one, i.e., the same viseme can appear for many different phonemes. This shows that visual information solely might not be enough for achieving the speech recognition task. As a result, VSR is challenging problem, specially, when using information only from plan marker-less and real-life images.

On the other hand, speaker identification is tightly coupled problem with speech recognition [4, 5, 3]. Speaker identification is defined as the ability to identify the speaker within a group of users from solely speech related features, like voice or mouth motion.

The appearance is not the ideal features to be used for solving the lipreading problem, while the dynamics in the utterance

¹ Viseme is the visual phoneme. It is defined as the smallest discriminative unit for visual speech

video attracts the researchers. Graphical models have been used extensively in VSR and AVSR. One technique that can be used to extract the dynamics in the video is Hidden Markov Model (HMM). HMM encodes the stochastic temporal relationship between sequence of observations [6]. In [7], HMM was used for encoding the visual dynamics of speech using Active Shape Model (ASM) and Active Appearance Model (AAM). A more general graphical models, Dynamic Bayesian Network (DBN) model, has been used in [8] with different visual articulation units called articulatory features. Graph embedding has been used in [9] to model the temporal relationship between frames, then the graph is used later, in [10], for estimating a curve that represent the dynamics in video. Graphical based methods try to capture the smooth temporal changes between the used visual units, but they may loose some visual information that may be crucial for discriminating small speech chunks like single letter utterance.

On the other hand, the work in [11] is based on extracting a single spatio-temporal feature vector for representing the visual and temporal information for the whole speech video. In [12] optical flow was used for extracting the whole word features. These two approaches outperform in the case of small size videos but it might be sensitive to frame outliers. Manifold-KPLS (MKPLS) framework is proposed in [13]. It finds a concise low-dimensional embedding for each visual unit. MKPLS has three phases: manifold parameterization, manifold latent space embedding and manifold classification. It uses Kernel-based supervised dimensionality reduction technique, namely, Kernel-PLS (KPLS) [14].

Choice of the used kernel in KPLS is very critical to achieve the best performance. This part has not been discussed in details in [13]. In this work, we explore and discuss several similarity measure for visual units. In MKPLS, visual units are represented by their manifold parameterizations. Even though kernel discussion is done from the point of view of MKPLS. Our findings can be generalized to be applied for other frameworks. We apply MKPLS for AVLetters [7] and OuluVs [11], and the experimental results contrast the performance variation between the kernels.

After this introduction, the problem statement, mathematical modeling and the MKPLS framework are presented in Section 2.2. Then, the used kernels are presented in-details in Section 3. Finally, the experimental results and the applied databases are listed in Section 4.

2. Manifold-KPLS

2.1. Problem Definition

Given set of visual units, we need to recognize new test unit, and infer the identity of the speaker. The visual unit could be viseme, word or a complete sentence. For each training visual unit is assigned to specific speech class and specific speaker.

Both the training and testing visual units are represented by sequence of images (frames) extracted from the speech video. Each frame exposes only the mouth area of the speaker.

2.2. Framework description

Manifold Kernel Partial Least Squares (MKPLS) framework is proposed in [13]. For convenience, we briefly describe the mathematical model and the framework pipelines here.

Let us denote the k -th sequence by $S_k = \{\mathbf{x}_i^k \in \mathbb{R}^D, i = 1 \dots n_k\}$, where the image $\mathbf{x}_i^k \in \mathbb{R}^D$. Let y_k represents the class labels for the k -th sequence. For the particular case of speech recognition and speaker identification, $y_k \in \{c_1, \dots, c_K\} \times \{p_1, \dots, p_L\}$. Here c_i is the speech label, and p_j is the performer identity. Let $\mathcal{M}_k \subset \mathbb{R}^D$ is a low-dimensional manifold connects the images of sequence S_k . The basic assumption is that all these manifolds ($\mathcal{M}_k \forall k$) are topologically equivalent, however each of them has different geometry in \mathbb{R}^D . This assumption is stated clearly in [13].

In principal, **MKPLS pipeline** has three phases: individual manifold parameterization, latent space embedding and finally inference/classification. Consider the manifold \mathcal{M}_k connects the n_k frames of specific Visual Unit (VU). Assuming, that we have a unified manifold \mathcal{U} . The result of **individual manifold parameterization** is a single representation for VU which holds the topological deformation of \mathcal{M}_k with respect to \mathcal{U} . Any further processing is done based on these parameterizations.

The manifold \mathcal{M}_k is represented by a parameterization \mathbf{C}_k with respect to a set of basis $\{\psi_1, \psi_2, \dots, \psi_n\}$. These basis are a nonlinear function of points on \mathcal{U} . We use Gaussian Radial Basis Function (Gaussian-RBF): $\psi_i(\mathbf{z}) = \exp(\sigma \|\mathbf{z} - \mathbf{w}_i\|)$ where $\mathbf{w}_i, i = 1 \dots n$ are fixed points on \mathcal{U} . The goal is to find a regression function $\gamma(t) = \mathbf{C}_k^\top \Psi(t)$ which minimize the objective function

$$\sum_i^{n_k} \left\| \mathbf{x}_i^k - \gamma^k(\mathbf{z}_i^k) \right\|^2 + \lambda \Omega[\gamma^k], \quad (1)$$

where $\|\cdot\|$ is the Euclidean norm, $\Psi(t) = [\psi_1(\mathbf{z}_t), \psi_2(\mathbf{z}_t), \dots, \psi_n(\mathbf{z}_t)]^\top$, Ω is a regularization function that enforces the smoothness in the learned function, and λ is the regularizer. Representer theory helps to find closed form for \mathbf{C} as

$$\mathbf{C}_k^\top = (\mathbf{A}_k^\top \mathbf{A}_k + \lambda \mathbf{G})^{-1} \mathbf{A}_k^\top \mathbf{X}_k^\top, \quad (2)$$

where \mathbf{A}_k is an $n_k \times n$ matrix with $\mathbf{A}_{(ij)} = \exp(\sigma \|\mathbf{z}_i - \mathbf{w}_j\|)$ and \mathbf{G} is an $n \times n$ matrix with $\mathbf{G}_{(ij)} = \exp(\sigma \|\mathbf{w}_i - \mathbf{w}_j\|)$. \mathbf{X}_k is the $n_k \times D$ data matrix for $\mathbf{U}\mathbf{V}_k$. The details of learning the manifold parameterization can be found in [13].

The choice of λ and n is crucial for better performance. Figure 1 shows the trade-off between value of λ and n . This choice depends upon the application. In this work, we need to capture the smooth dynamics in the visual units. Therefore, we choose $\lambda = 50$. It is clear that $n = 16$ expose more variations than with $n = 8$. More information can be useful in some cases and can be more confusing in others. Therefore, in Section 4, we show both configurations.

In **Latent space embedding**, kernel partial least squares (KPLS) [14] is adopted for embedding the parameterizations $\{\mathbf{C}_k, k = 1 \dots N\}$ into a low-dimensional latent space \mathbb{R}^m , as $\{\mathbf{t}_k \in \mathbb{R}^m, k = 1 \dots N\}$. KPLS is supervised method, so it uses the set of labels $\{y_k, k = 1 \dots N\}$ for the embedding. Supervised embedding guarantees to achieve the most concise and informative low-dimensional latent space embedding. For

any manifold \mathcal{M}_ν , represented by its parameterization \mathbf{C}_ν , the corresponding embedded point can be computed by

$$\mathbf{t}_\nu = \mathbf{v}_\nu \mathbf{R}. \quad (3)$$

Where the projection matrix \mathbf{R} is learned from KPLS algorithm, and $\mathbf{v}_\nu = K(\mathbf{C}_\nu, \cdot) \in \mathbb{R}^N$. K measures the similarity between \mathbf{C}_ν and all training manifold parameterizations $\{\mathbf{C}_k, k = 1 \dots N\}$. The choice of kernel K and its computation is discussed in detailed in Section 3. Because we solve two problems, speech recognition and speaker identification, we learn one \mathbf{R} for each task. We learn \mathbf{R}_c based on speech labels, and we learn \mathbf{R}_p for speaker identification with subject labels. As a result, for each \mathbf{v}_ν , we get two embedding in two latent spaces: $\mathbf{t}_\nu^c = \mathbf{v}_\nu \mathbf{R}_c$ in the speech latent space and $\mathbf{t}_\nu^p = \mathbf{v}_\nu \mathbf{R}_p$ in the speaker latent space.

Finally in **manifold classification**, given a latent space embedding \mathbf{t}_ν , MKPLS uses several techniques to classify it such as Regression for classification (RfC) [13], Support vector machines (SVM) and K-nearest neighbor (KNN). In the latent space of speech, we want to infer the speech label (c) while in the speaker space, we need to infer the subject label (p).

3. Manifold-to-manifold Kernels

The parameterization, extracted out of the first phase of MKPLS, holds the dynamics in each video which encodes speech-related information along with speaker-related information. Because MKPLS uses kernel-based approach for dimensionality reduction, the kernel choice is critical for achieving the best performance. In this section, we investigate several types of kernels.

MKPLS claims that, to define manifold-to-manifold kernel, it suffices to define it in the parameterization space, i.e., $K_{manifold}(\mathcal{M}_i, \mathcal{M}_j) \doteq K(\mathbf{C}_i, \mathbf{C}_j)$. Therefore, we need to define kernels over the space of parameterizations, which consequently, measure the similarity between manifolds in terms of their geometric deformation from the common manifold representation. MKPLS gives us the ability to plugin any valid kernel. In this section, we investigate several choice of kernels: matrix-based kernels, curve-based kernels and subspace-based kernels.

3.1. Matrix-based kernels

Since the dimensionality of all parameterizations is unique, we can measure the similarity between them by measuring the similarity between the corresponding column. This is the idea behind the matrix-based kernels.

3.1.1. Cosine-similarity kernel (Cosine)

We can measure the similarity between columns using cosine the angle between them. As a result, the overall similarity between two parameterizations is the sum over all column-wise similarities. Therefore, the cosine-manifold kernel can be defined as

$$K_{\cos}(\mathbf{C}_i, \mathbf{C}_j) = \frac{\text{tr}(\mathbf{C}_i \mathbf{C}_j^\top)^2}{\|\mathbf{C}_i\|_F \|\mathbf{C}_j\|_F}, \quad (4)$$

where $\|\cdot\|_F$ is matrix Frobenius norm.

3.1.2. Euclidean-distance kernel (Eculid)

In this kernel, we measure the Euclidean distance between the i -th column in parameterization \mathbf{C}_1 (u_{1i}) and its corresponding

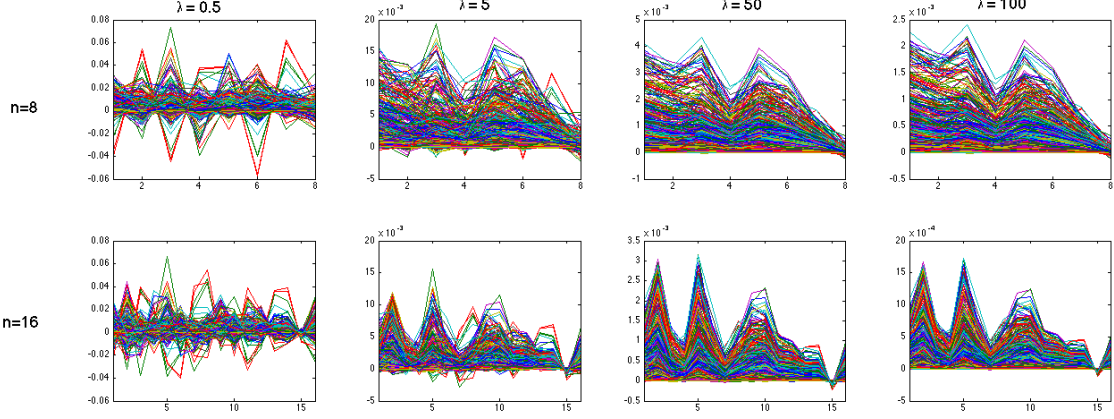


Figure 1: The parameterization C is $D \times n$ matrix. Each plot has D lines, and each line is a plot for values progression of a row in C . At large values of λ the parameterization is smooth enough to capture large dynamics in the visual unit.

Table 1: Subject Semi-dependent speech recognition on **OuluVs** database

$m =$	$n = 8$ and $1 \times 2 \mathbf{LBP}_{1-8 \times 8}^{u_2}$							$n = 16$ and $1 \times 1 \mathbf{LBP}_{1-8 \times 8}^{u_2}$						
	10	30	50	80	100	130	200	10	30	50	80	100	130	200
Cosine	62.19	78.13	81.72	81.41	81.72	82.03	81.09	58.28	77.19	79.22	79.53	79.22	79.84	79.53
Euclid	61.25	79.06	79.38	79.53	79.84	80.16	78.91	56.72	75.16	75.00	75.63	75.94	75.31	74.69
EditDist	62.50	75.63	66.72	43.44	22.81	21.41	22.34	59.53	70.16	61.25	41.25	35.00	24.69	15.62
Frechet	29.53	27.81	25.47	17.34	15.97	14.53	11.88							
Grassm	28.91	37.34	41.87	42.19	39.69	37.66	31.88	24.38	26.41	29.53	28.44	26.25	25.63	25.63
GrassmCC	28.91	37.34	41.87	42.19	39.84	37.34	27.81	24.53	26.41	29.53	28.44	25.94	25.31	21.72
GrassmDiff								28.13	35.00	37.81	39.17	37.29	31.67	25.63

column in parameterization $C_2(u_{2i})$. Hence, the overall matrix kernel $\delta = \sum_{i=1}^n \|u_i - v_i\|_2^2$, and the matrix similarity is

$$K(C_1, C_2) = \exp(-\omega\delta) \quad (5)$$

where ω is a normalization factor. For $K(\cdot, \cdot)$ to be valid kernel, it needs to be symmetric positive definite matrix (SPD). The exponential function takes care of the positive definiteness part. For the symmetry, the used distance measure should be metric, which is satisfied for Euclidean distance case.

3.2. Curve-based Kernels

In this category, we consider the columns of the parameterization matrix as points in \mathbb{R}^D , and the matrix defines a curve connecting those points. The matching between columns should obey the ordering. This means that if two columns i, j from the first matrix match the columns u, v from other matrix respectively, the $u \leq v$ iff $i < j$. For each of the following distances, the parameterization kernel is computed using Eqn 5.

3.2.1. Fréchet-distance Kernel (Frechet)

Fréchet distance is a known metric to measure the distance between two curves, that takes into account the location and ordering of the points along the curves. Here, we use discrete Fréchet distance, also known as coupling distance, in which we assume that the curves are piece-wise linear. The basic idea that, each point in one curve is matched with its closest point on the other curve, and the distance d will be the maximum Euclidean distance between each two matched points. At the end, not all points are matched between the two curves.

3.2.2. Edit-distance kernel (EditDist)

The idea of this metric is similar to minimum edit distance between strings. Two main difference between EditDist and Frechet algorithms: in EditDist the overall distance is the sum of distance between all matches while Frechet takes the maximum of all matches, and EditDist considers unmatched points as being matched with the origin while Frechet ignores the unmatched points. Emperically, we found that the best column-wise distance, in both EditDist and Frechet, is the Euclidean distance.

3.3. Subspace-based Kernel

Each parameterization C_k represents n -dimensional subspace in \mathbb{R}^D . Therefore, we can use subspace-to-space metric to measure the similarity in parameterization space. This gives the most general comparison between matrices. Because it considers the subspace spanned by the columns of each parameterization without encoding any ordering.

3.3.1. Grassmannian kernel

Every coefficient matrix C_k is $D \times d$. Since $D \gg d$, hence C_i represent d dimensional subspace in \mathbb{R}^D . Therefore, the matrix C belongs to Grassmannian manifold $G_{D,d}$. For more details about Grassmannian manifolds, the reader is referred to [15].

There are several approaches for measuring the similarity on Grassmannian manifold, we use the one defined in [16].

$$K_{ij} = a_1 K_{ij}^{cc} + a_2 K_{ij}^{proj} \quad (6)$$

Where K_{ij}^{proj} , K_{ij}^{cc} are the projection kernel and the canon-

Table 2: Speaker Independent - speech recognition Accuracy on **OuluVs** database

$m =$	$n = 8$ and $1 \times 2 \text{LBP}_{1-8 \times 8}^{u_2}$							$n = 16$ and $1 \times 1 \text{LBP}_{1-8 \times 8}^{u_2}$						
	10	30	50	80	100	130	200	10	30	50	80	100	130	200
Cosine	49.22	50.00	51.41	49.84	50.00	49.84	50.00	50	51.88	49.06	49.06	50.31	50.00	50.63
Euclid	48.59	55.16	55.78	55.00	55.47	55.47	55.00	48.44	61.25	58.44	57.50	57.81	58.75	57.19
EditDist	47.97	50.94	41.25	26.09	21.72	18.13	16.41	48.44	48.44	43.75	37.81	31.56	21.87	16.25
Frechet	11.25	13.28	12.50	12.34	13.75	13.44	13.75							
Grassm	29.84	30.31	32.34	31.09	28.59	24.84	31.56	22.19	23.75	25.00	19.69	21.25	19.06	15.31
GrassmCC	29.69	30.31	32.34	31.09	28.75	23.91	23.28	22.19	23.75	25.00	19.69	21.25	19.37	17.19

ical correlation kernel respectively, and a_1, a_2 are weighting constants. The projection kernel is defined by $\mathbf{K}_{ij}^{proj} = \|\Pi_i^\top \Pi_j\|_F^2$ where Π_k is the orthogonal version of the coefficient matrix \mathbf{C}_k , computed by Gram-Schmidt orthogonalization algorithm. The canonical correlation kernel is defined by

$$\mathbf{K}_{ij}^{cc} = \max_{a_p \in \text{span}\{\Pi_i\}} \max_{b_q \in \text{span}\{\Pi_j\}} a_p^\top b_q \quad (7)$$

Subject to $a_p^\top a_q = b_p^\top b_q = 1$ if $p = q$, and 0 otherwise. We use two Grassmannian kernels: **Grassm** defined by Eq 6 and **GrassmCC** defined by Eq 7.

Since Grassmannian distance does not consider the ordering of the parameterization columns, we can encode some temporal information by using the parameterization of difference of the input features. We denote this experiment by **GrassmDiff**.

4. Experiments

4.1. Databases

We apply MKPLS for OuluVs database [11]. OuluVs has ten different everyday phrases. Each phrase is uttered by 20 subjects up to five times. We use the same test protocol used in [13]. The frame rate was set to 25 fps. The dataset contains sequence of images for mouth area with average resolution of 120×60 pixels. **LBP** [17] visual features is extracted from images. Two feature configurations have been used: the first configuration is **LBP**_{1:8×8} with $n = 16$ (16 basis for Gaussian-RBF) and the second one is $1 \times 2 \text{LBP}_{1:8 \times 8}^{u_2}$ with $n = 8$. We also apply MKPLS for AVLetters database [7] which has ten subjects. Each speaker repeats every English letter ($A \cdots Z$) exactly three times, with a total of 780 video sequences. The speaker was requested to start and end utterance of every letter in a neutral state (mouth closed). We apply single feature configuration: 3×4 cell-grid with four-resolutions LBP features ($3 \times 4 \text{LBP}_{1:4 \times 8}^{u_2}$) with $n = 8$.

In all experiments, the recognition rate is measured as the ratio between the correctly recognized clips and the total number of clips.

4.2. Experimental Results

In this section, we present the empirical results of MKPLS when plugged with each one of the kernels described in Section 3. To give real comparison between the proposed kernels, we need to explore different parameters of MKPLS pipeline: The number (n) of Gaussian-RBF basis ψ that we use to learn the individual manifold parameterization, we show results for $n = 8, 16$. The dimensionality of the manifold latent space, we use $m = 10, 30, 50, 80, 100, 130, 200$, which cover wide range of the possible values.

Especially for Grassmann-based kernels 3, we show the affect of using the parameterization of the LBP of the images it-

self vs LBP of the images concatenated with parameterization of the discrete difference between those images.

4.2.1. Visual speech recognition

Two test protocols has been adopted for visual speech recognition: Speaker Independent (SI) and Speaker Semi-Dependent (SSD) as defined in [13].

Speaker Semi-Dependent VSR (SSD): Here we test on one repeat of the available videos and train based on the remaining repeats for the same subjects. In this configuration all subjects and phrases are presented in the training set. Table 1 show the SSD speech recognition accuracy for OuluVs database with the two feature configurations. Table 4 shows the result of matrix-based kernels and curve-based kernels applied avletters.

Table 4: SSD speech recognition on **AvLetters**

$m =$	$n = 8$ and $3 \times 4 \text{LBP}_{1:4 \times 8}^{u_2}$						
	10	30	50	80	100	130	200
Cosine	50.77	56.67	60.77	62.82	63.85	64.49	63.85
Euclid	51.41	56.41	60.38	64.49	65.13	64.52	64.74
EditDist	51.41	56.54	60.51	64.36	65.13	65.00	64.74
Frechet	23.59	34.62	36.28	34.49	35.64	34.49	29.23

Speaker Independent VSR (SI): the challenge here is to show the scalability of the model, mean how far the model can recognize the spoken phrase based on the dynamics even if the speaker is not seen before in the training set. In this experiment, we use one-speaker-out. Table 2 show the SI speech recognition accuracy for OuluVs for the two configurations.

4.2.2. Speaker Identification (SpId):

The goal in this experiment is to find the speaker within the register set of users. The challenge is to find the speaker from the limited available information in the mouth area. We take one repetition out for testing, and train over all other repetitions. Table 3 shows the speaker identification accuracy when applied to OuluVs for the two test configurations.

4.3. Discussion

From the numbers, we can clearly see the superiority of both techniques of Matrix-based kernels (Cosine and Euclid) in all test and features configurations. For $m = 10$, EditDist gives the best results for SSD-speech recognition and Speaker Identification. GrassDiff gives slightly better results than Grassm and GrassCC, since it encodes temporal information. Frechet kernel proves failure in this application. For Avletters, EditDist and Euclid give the best recognition rate.

Table 3: Speaker Identification Accuracy on OuluVs database

$m =$	$n = 8$ and 1×2 $\mathbf{LBP}_{1-8 \times 8}^{u_2}$							$n = 16$ and 1×1 $\mathbf{LBP}_{1-8 \times 8}^{u_2}$						
	10	30	50	80	100	130	200	10	30	50	80	100	130	200
Cosine	93.91	99.69	99.69	99.69	99.69	99.69	99.69	92.66	99.69	99.69	99.69	99.69	99.69	99.69
Euclid	93.75	99.53	99.53	99.53	99.53	99.53	99.53	93.91	99.53	99.53	99.53	99.69	99.53	99.53
EditDist	94.22	99.53	99.53	99.37	99.06	71.88	69.69	92.81	99.69	99.53	99.53	99.53	99.69	99.53
Frechet	83.91	95.16	89.06	75.94	64.84	50.63	17.97	88.13	96.09	87.81	62.81	27.66	27.66	27.66
Grassm	84.69	99.38	99.37	99.53	99.06	98.75	97.81	92.19	99.22	99.06	98.91	98.44	98.33	97.56
GrassmCC	84.69	99.38	99.37	99.53	99.06	98.75	97.81	92.19	99.22	99.06	98.91	98.44	98.33	95.94

5. Conclusion

We investigated the kernel choice for the middle phase of MKPLS framework. We explored three kind of manifold kernels: matrix-based kernel, curve-based kernel and subspace-based kernel. We compare the kernels based on different parameter configuration for MKPLS. The experiments shows that using parameterization-to-parameterization kernel can delegate manifold-to-manifold kernel. The results shows the superiority of the matrix-based kernel for visual speech recognition. For speaker identification tasks, all kernels gives perfect results.

6. References

- [1] G. Potamianos and C. Neti, "Audio-visual automatic speech recognition: An overview," *Issues in Visual and Audio-Visual Speech Processing*.
- [2] H. McGurk and J. MacDonald, "Hearing lips and seeing voices," *Nature*, vol. 264, no. 23 December, pp. 746–748, 1976.
- [3] D. Shiell and L. Terry, "Audio-Visual and Visual-Only Speech and Speaker Recognition: Issues about Theory, System Design, and Implementation," *Visual speech recognition: lip segmentation and mapping*.
- [4] J. Luettin, N. Thacker, and S. Beet, "Speaker identification by lipreading," *International Conference on Spoken Language Processing*, pp. 1–4.
- [5] C. Sanderson and K. Paliwal, "Identity verification using speech and face information," *Digital Signal Processing*, no. 5, pp. 449–480, Sep.
- [6] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, no. 2, pp. 257–286.
- [7] I. Matthews and T. Cootes, "Extraction of visual features for lipreading," *PAMI*, vol. 24, no. 2, pp. 198–213, 2002.
- [8] K. Saenko and K. Livescu, "Visual speech recognition with loosely synchronized feature streams," *IEEE International Conference on Computer Vision*.
- [9] Z. Zhou and G. Zhao, "Lipreading: A Graph Embedding Approach," *ICPR*, pp. 523–526, 2010.
- [10] Z. Zhou, G. Zhao, and M. Pietikainen, "Towards a practical lipreading system," *Computer Vision and Pattern Recognition*, 2011.
- [11] G. Zhao, "Lipreading with local spatiotemporal descriptors," *IEEE Transactions on Multimedia*, pp. 1–11, 2009.
- [12] A. Shaikh, D. Kumar, and W. Yau, "Lip Reading using Optical Flow and Support Vector Machines," *IEEE International Congress on Image and Signal Processing*, vol. 1, pp. 327–330, Oct. 2010.
- [13] A. Bakry and A. Elgammal, "MKPLS: Manifold Kernel Partial Least Squares for Lipreading and Speaker Identification," *Computer Vision and Pattern Recognition*, 2013.
- [14] R. Rosipal and L. Trejo, "Kernel partial least squares regression in reproducing kernel hilbert space," *The Journal of Machine Learning Research*, pp. 97–123.
- [15] A. Edelman, T. a. Arias, and S. T. Smith, "The Geometry of Algorithms with Orthogonality Constraints," *SIAM Journal on Matrix Analysis and Applications*, no. 2, pp. 303–353, Jan.
- [16] M. Harandi and C. Sanderson, "Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2705–2712, Jun.
- [17] T. Ojala, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *PAMI*, no. 7, pp. 971–987.